Unbiased Anomalous Trajectory Detection With Hierarchical Sequence Modeling

Xiangjie Kong, *Senior Member, IEEE*, Yuwei He, Guojiang Shen, Jiaxin Du, Zhi Liu, and Ivan Lee

***Abstract*—Anomalous trajectory detection plays an important role in the field of trajectory big data mining, providing significant support for identifying drivers traveling at inappropriate speeds and detecting cab fraud. Current studies often use equal-sized grids to represent trajectory points, and they mainly focus on the general shape of trajectories while ignoring the spatial density distribution of trajectories. In addition, existing generative models are biased in learning the patterns of normal trajectories, and the same bias exists in processing labeling information. To address the above two problems, we propose an unbiased anomalous trajectory detection method (HS-UATD) based on hierarchical sequence modeling. Our method constructs a hierarchical structure of the entire spatial region using a quadtree, which captures the location density distribution of the entire spatial region. Our model captures the rich spatio-temporal pattern of trajectories containing spatial hierarchical information and learns the probability distribution of unbiased normal trajectories. We employ both clustering algorithms and anomaly injection techniques to obtain unbiased labeling information, and we define trajectories that deviate from the normal pattern as anomalies. Through extensive experiments on three unbiased, biased and real trajectory datasets, we validate the effectiveness of the method.**

***Index Terms*—Anomalous trajectory detection, hierarchical structure, graph transformer, generative model, normal trajectory.**

I. Introduction

THE DEVELOPMENT of Internet of Things (IoT) technology has greatly advanced the development of urban intelligent transportation systems, where sensor networks, including GPS, are combined with Internet of Things technology. The spatio-temporal trajectories of a large number of moving objects are collected, and these trajectories data contain rich information about traffic flow and people, so analyzing these trajectories data [1], [2], [3], [4], [5], [6], [7] has become a popular research area. Among them, vehicle anomalous trajectory detection is a topic of interest in real-world scenarios, helping us to identify cab driving frauds and some inappropriate speed driving, which improves the travel experience of residents and strengthens the public safety management of the city in the intelligent transportation system.

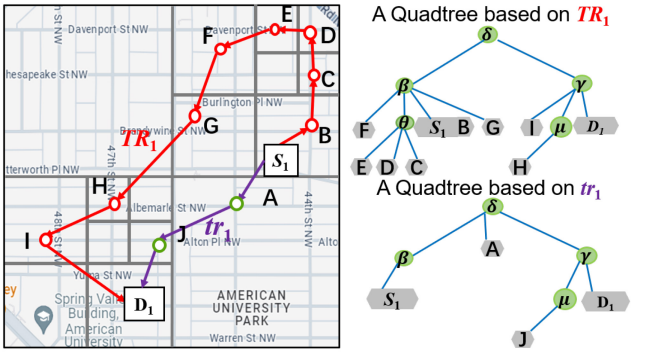


Fig. 1. The trajectory point E, D, C in trajectory TR_1 have a higher density in the spatial distribution and are located at a lower level at hierarchical quadtree on the right, while the D_1 point has a lower density and is located at a higher level at hierarchical quadtree on the right. And the density distribution, i.e., hierarchical structure, of the anomalous trajectory TR_1 is relatively different from that of the normal trajectory tr_1.

In recent years, anomalous trajectory detection has attracted extensive research attention. Researchers have done a lot of researches and proposed many methods. Early studies employ manually designed trajectory features using statistical analysis [8], [9], [10], [11], [12] and machine learning [13], [14] methods to calculate the spatial similarity between the target trajectory and the normal trajectory. With the development of deep learning models [15], [16], [17], [18], [19], anomalous trajectory detection methods that capture sequence information are proposed, which are more efficient and accurate than traditional methods. Existing anomalous trajectories can be categorized into vehicle-based anomalies and driver-based anomalies [20]. We focus on driver-based anomalies. The driver-based anomalies in this paper are detour anomalies and speed anomalies, which are caused by driver behavior, i.e., spatial anomalies and temporal anomalies as mentioned below. Most existing supervised detection methods can only detect predefined types of anomalies. However, as mentioned above, the variety of vehicle anomalies is so large that it may be impractical to develop a comprehensive set of rules that encompasses all possible anomalies. Therefore we need to build a model that can recognize various anomalies through unsupervised learning.

Received 6 June 2024; revised 5 October 2024 and 11 December 2024; accepted 25 December 2024. Date of publication 30 December 2024; date of current version 12 June 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62476247, Grant 6207395, and Grant 62072409; in part by the "Pioneer" and "Leading Goose" Research and Development Program of Zhejiang under Grant 2024C01214; and in part by the Zhejiang Provincial Natural Science Foundation under Grant LR21F020003. *(Corresponding author: Guojiang Shen.)*

Xiangjie Kong, Yuwei He, Guojiang Shen, Jiaxin Du, and Zhi Liu are with the College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China (e-mail: xjkong@ieee.org; 211122120044@zjut.edu.cn; gjshen1975@zjut.edu.cn; jiaxin.joyce.du@gmail.com; lzhi@zjut.edu.cn).

Ivan Lee is with the UniSA STEM, University of South Australia, Adelaide, SA 5001, Australia (e-mail: Ivan.Lee@unisa.edu.au).

Digital Object Identifier 10.1109/TCE.2024.3523565

1558-4127 © 2024 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.
Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.

However, this poses a great challenge to our work since existing anomalous trajectory detection methods face several problems as follows:

- **Limited learned information.** Existing works learn limited trajectory information and struggle to capture rich spatio-temporal patterns. Most grid-based anomalous trajectory detection methods often use equal-sized grids [8], [9], [17], [18], [19], [21], [22] to represent the trajectory points, which cannot learn other spatial distribution patterns of the trajectory, such as hierarchical structure information that reflects the spatial density distribution. Common RNNs that capture sequence information also suffer from missing information when dealing with long trajectories.
- **Biased datasets.** Some previous unsupervised detection methods learn patterns of normal trajectories in biased datasets. Most of the works directly use manual labeling [8], [9] or anomaly injection [18], [19], [21] when dealing with the labeling information of the datasets. The use of manual labeling of data has laborious and credibility issues, while anomaly injection into the dataset needs to be done in a clean dataset, and if anomalous injection is done in a dataset with anomalous trajectories (realistically unprocessed data), it introduces bias.

To address the Limited learned information challenge, we adopt an innovative approach to obtain spatial hierarchical information of trajectories by constructing a quadtree to map trajectories to unequal-sized grids, which captures the positional density distribution of the entire spatial region. As shown in Fig. 1, this hierarchical information is crucial for understanding the spatial distribution pattern of trajectory data. Because regions with different spatial hierarchies have different trajectory densities and characteristics, the trajectory data are spatially unevenly distributed, i.e., some regions may contain trajectory points more densely than others. In dense trajectory regions, which are smaller grid cells in order to represent the spatial distribution of trajectories in more detail, they are at a lower level. In sparse regions, they are larger grid cells to reduce computational complexity and increase efficiency, and are located at higher levels. By using different scales of grids at different levels and dynamically adjusting the grid size according to the density distribution of the trajectory data, the characteristics of the trajectory data can be better adapted to different scenarios. In order to obtain the hierarchical information of trajectories to enhance trajectory embedding, we perform Node2Vec on the quadtree. Then the graph is constructed based on the quadtree, which captures the hierarchical information of the trajectory. Currently, the RNN model is often used to capture the relationship between sequences, but it is very time-consuming to train and has the problem of missing information when dealing with long trajectories. Considering the spatial information, we use Graph transformer to infer the trajectory graph and capture the dependency between the trajectory sequences as well as the spatial hierarchical structure of the trajectories. Thus, the challenge of limited learned information is addressed.

In order to solve the biased datasets challenge, normal trajectories are obtained with the clustering algorithm [23] to get an unbiased training set. The training phase is to learn the pattern of normal trajectories, so it is necessary to have a clean training set. And we process the labeling information of the dataset with anomaly injection on the normal trajectories obtained with the clustering algorithm to get an unbiased testing set.

In this paper, we propose a novel unsupervised anomalous trajectory detection method based on hierarchical sequence modeling (HS-UATD), which is a method that recognizes anomalies effectively. To summarize, the main contributions of this paper are as follows:

1) The proposed technique is the first time that trajectories are automatically mapped to unequal-sized grids with attention to the spatial density distribution of the trajectories, which allow for finer-grained analysis and processing to improve detection accuracy.
2) We use quadtrees to construct trajectory graphs and Graph Transformer as an inference network, thus capturing the spatial hierarchical relationships of trajectories and the complete trajectory sequence information, which captures the rich spatio-temporal patterns of trajectories in the inference network part.
3) Our model is extensively experimented on three unbiased and biased datasets. The experimental results show that our method generally outperform existing SOTA methods and demonstrate the importance of unbiased datasets. In addition, this paper characterizes the real-time challenges and future perspectives of anomalous trajectory detection, pointing the way to the next step of research.

The rest of the paper is organized as follows: we present related work in Section II. In Section III, we introduce preliminary. In Section IV, we describe model in detail. In Section V, we verify the effectiveness of our algorithm. The last section summarizes and discusses the paper.

II. Related Work

A. Anomalous Trajectory Detection

Existing research favors driver-based anomalous trajectories. We categorize existing abnormal trajectory detection algorithms into two groups: metric-based and learning-based. Metric-based methods often compare the target trajectory with a customized normal trajectory based on metrics such as distance or density, i.e., anomaly judgments are made based on predefined thresholds. For example, Zhang et al. [8] proposed an anomaly detection framework called IBAT, which detects anomalies by comparing the degree of isolation of other trajectories. This is a classic method that laid the foundation for many later works. Wang et al. [9] proposed an anomaly detection framework called ATDC, which is a method for detecting and classifying anomalous trajectories based on the difference and intersecting distances. There are some studies [11], [12] incorporating other metrics. However, all metric-based methods have the following drawbacks: heuristic definitions (i.e., thresholds) of their normal trajectories are usually hand-crafted, which leads to time-consuming computation of thresholds for normal trajectories. In addition, the reliability and robustness of metric-based methods turn out to depend

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.

mainly on the setting of these threshold parameters, which is unreasonable.

With the development of deep learning techniques, some studies in recent years have been based on learning for abnormal trajectory detection. A previous study [23] proposed a vehicle anomaly detection method called DB-TOD, which is based on a probabilistic model to model the driving preferences of a historical trajectory set. However, methods based on metrics and probabilistic learning do not characterize the sequential information of trajectories well, and these methods mainly focus on sub-trajectories or trajectory points while ignoring the effect of the whole trajectory sequence information. Cheng et al. [17] proposed a model called STRNN to capture spatio-temporal correlations in trajectories. However, this approach relies on supervised learning and requires considerable manual effort to recognize normal or abnormal trajectories before training. Therefore, some unsupervised learning models have been proposed in recent years. For example, GM-VSAE [18] is an RNN-based variational autoencoder that generates the likelihood that a target trajectory belongs to a normal trajectory by learning the probability distribution of the normal trajectory. Wang et al. [19] proposed UA-OATD, which is an improved version of GM-VSAE that considers the grid's neighbor information and introduces an attention mechanism to encode the trajectories. However, these unsupervised learning models learn a single trajectory information and are unable to learn the spatial hierarchical information of trajectories and the complete sequence dependencies of longer trajectories.

B. Generative Anomaly Detection

Reconstruction-based Generative trajectory anomaly detection methods has been widely used for anomaly detection and localization. The generative model learns a good representation of normal samples by optimizing learning in a training set with only normal data. The generative model reconstructs the input samples and uses the reconstruction error to determine the anomalies. Malhotra et al. [24] proposed an LSTM-based auto-encoder(AE) for anomaly detection based on the reconstruction error. The main role of the AE is to obtain low-dimensional hidden vectors by dimensionality reduction. Whereas, the VAE combines the ideas of auto-encoder and probabilistic modeling, which adds noise to the hidden vectors, making the hidden vector's coding region from discrete to continuous, which enlarges the coding region and makes the generation better. References [18], [19], [21] use an RNN-based VAE model to encode trajectories as vectors in the latent space and learns their distributions. GANs also perform well in generating realistic and lifelike data samples, because their generation process involves a game process between a generator and a discriminator, which makes the generated samples closer to the real distributions. Deng et al. [25] proposed a graph convolution-based Adversarial Network for spatio-temporal anomaly detection. However, the training process of GAN is unstable, which sometimes causes the generator to crash or the quality of generated samples to degrade. Some existing works [26], [27], [28] use Diffusion Model for anomaly detection, because Diffusion Models perform better in generating high-quality and realistic samples when generating high-dimensional data such as images. However, Diffusion Models are computationally expensive. After comprehensively considering the characteristics and dimensions of trajectory data, we choose VAE Model as our base model.

C. Graph Transformers

Existing works [29] commonly use RNN to encode trajectories, but they are unable to capture the full sequence dependencies of longer trajectories. And Transformer based on attention mechanism can solve this limitation, so [22] uses an upgraded version of Transformer, UT-Transformer, to learn the trajectory representation. However, Transformer does not fulfill the requirement when spatial structure information is considered. Many works [30], [31], [32] combine graph neural networks and Transformer to capture the spatial structure information of sequences. Graph Transformer was born. Li et al. [32] proposed a model that focuses on all the nodes in the graph instead of the local neighbors of the nodes in order to capture the global information of the graph. This limits the effective utilization of sparsity. It is highly undesirable to give up sparsity and local context in order to obtain global information about the graph, but there are other ways to achieve the same goal, such as using node Laplacian positional feature vectors [33] or graph-specific positional features [34]. Therefore, Dwivedi and Bresson [35] proposed a generic Graph Transformer. In this model, the attention mechanism is a function of the neighborhood connectivity of each node in the graph, the position encoding is represented by Laplacian feature vectors; and the batch normalization layer is used instead of the layer normalization, which provides better generalization performance and faster training. Finally, edge feature representation is also considered in this architecture. In order to capture the spatio-temporal patterns of the vehicle trajectories, graph transformer is used to encode the trajectories in this paper.

III. Preliminary

A. Problem Definition

Definition 1: Raw Trajectory Point. When a GPS takes a sample, the record information is called the raw trajectory point p, and it is represented by $p = (lon, lat, time)$, where (lon, lat) is the longitude and latitude of the spatial position, and time is the moment when the GPS records the position information.

Definition 2: Raw Trajectory Set. We define a raw trajectory set T_{set} from a source S_T to a destination D_T, $T_{set} = \{Tr_1, Tr_2, Tr_3, \ldots, Tr_{len(T_{set})}\}$, where each trajectory Tr_i is a sequence of raw trajectory point. $Tr_i = \{p_1, p_2, p_3, \ldots, p_{len(Tr_i)}\}$, where p_j is the j-th point in Tr_i.

Problem Definition: Anomalous Trajectory Detection. Anomalous trajectories can be defined as movement patterns that significantly deviate from the normal model or desired behavior in space and time. Specifically, a trajectory can be judged as anomalous when it significantly deviates from the usual clusters or major paths in spatial terms, e.g., detours; its temporal behavior does not match the normal model, e.g., inappropriate speeds; and the frequency performance is

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.

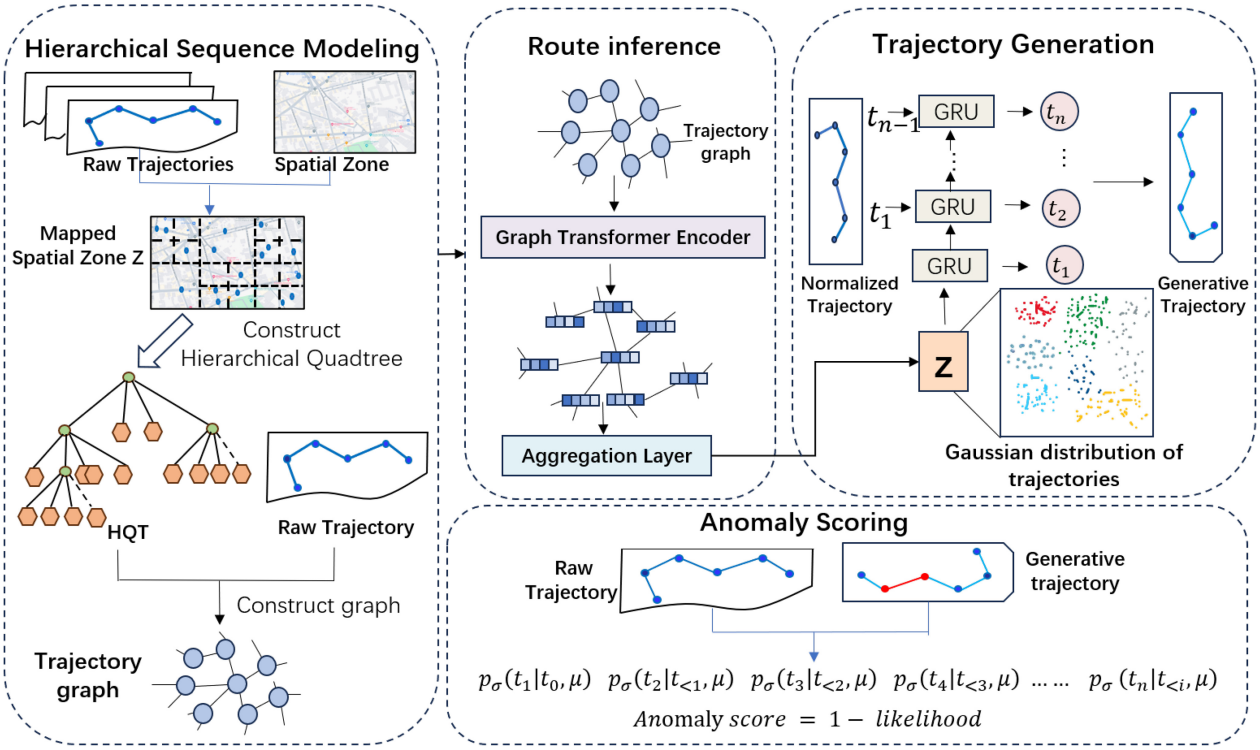


Fig. 2. The architecture of the proposed HS-UATD.

significantly different from the expected behavior. This definition allows for more precise capture of anomalous trajectories based on specific application scenarios.

B. Overview of HS-UATD

As shown in Fig. 2, our model consists of four parts, hierarchical sequence modeling, route inference, trajectory generation, and anomaly scoring. In the first part, we construct a quadtree for the whole region, and the trajectories are mapped to unequal-sized grids, thus reflecting the density distribution of the whole region. we perform Node2Vec on the quadtree to obtain the trajectory hierarchical information. Based on the quadtree, we construct a graph for each trajectory, and select nodes and connect edges according to special rules, thus constructing a trajectory graph that can reflect rich spatial patterns. In the second part, we get an input embedding for each input trajectory graph after the first part, which is encoded into vectors in continuous latent space using Graph transformer and aggregation layer. Then the vectors of trajectories in continuous latent space are modeled with Gaussian model. The distribution describes the probability of normal distribution of all trajectories. In the third part, we sample the trajectories from the Gaussian distribution and then generate them from the GRU network. In the fourth part, the trajectory anomaly scores are computed by likelihood of trajectories.

IV. Methodology

A. Hierarchical Sequence Modeling

In this section, we will detail how to convert a historical trajectory into a trajectory graph.

1) Hierarchical Quadtree Construction: HS-UATD models the spatial hierarchical information using quadtree [36]. It recursively divides the zone into four equal quadrants, subquadrants, and so on, as shown in Fig. 3. For the given historical trajectory set, we project their positional records to region $\mathcal{Z}$ and then recursively decompose the region, and treat subregions as child nodes until no leaf node's position exceeds a threshold η. Thus a constructed hierarchical quadtree $\mathcal{HQT}$ of trajectories is constructed. An example of a constructed hierarchical quadtree $\mathcal{HQT}$ with $\eta = 2$ is shown in Fig. 3. The choice of η is described later. The choice of constructing a quadtree is due to its ability to recursively divide the space into different regions based on how dense or sparse the data is. The natural hierarchical structure of the quadtree helps the model to capture information about the density distribution of the trajectories, allowing the spatial relationships of the trajectories to be expressed hierarchically from coarse to fine.

2) Node Embeddings in Quadtree: We learn the embeddings of grid nodes in $\mathcal{HQT}$ in order to fuse spatial hierarchical information in trajectory coding, i.e., enhanced trajectory coding. Thus, we firstly pre-train the $\mathcal{HQT}$ to obtain the embeddings of grid nodes. In this process, we consider the $\mathcal{HQT}$ as a graph and employ the Node2Vec algorithm. Specifically, the Node2Vec algorithm implements the learning of vector representations of the nodes in the graph through a combination of random wandering and Skip-gram modeling. In the random walk phase, a multi-step random walk is performed starting from each node, and multiple node sequences are generated by controlling the probability of entering the previous node and jumping to neighboring nodes. These sequences are used as training data and are trained by Skip-gram model with the goal of predicting

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.

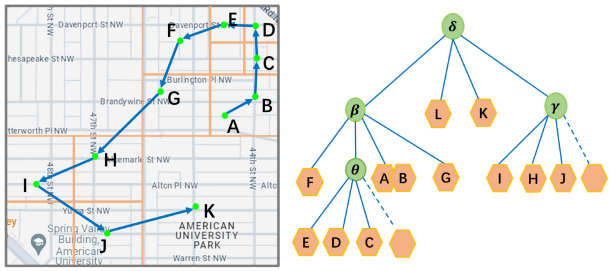


Fig. 3. A example of quadtree construction: On the left, we project positional records of a trajectory to region $\mathcal{Z}$. On the right, a hierarchical quadtree based on $\mathcal{Z}$.

the current node by the neighborhood information of the node. Eventually, each node is given a fixed-dimension vector representation that captures its contextual information in the graph structure, making similar nodes closer in the vector space. We obtain an embedding matrix $E_T \in d_T \times N_T$ that captures the hierarchical information of the $\mathcal{HQT}$, where d_T is the number of embedding dimensions of the grid nodes. Node2Vec performs embedding learning on this hierarchical structure, which is able to better extract information about the density distribution of trajectory points and preserve rich spatio-temporal patterns.

3) Construct History Trajectory Graph: Based on the given trajectories T and $\mathcal{HQT}$, we construct a graph $G_T = (N, E)$. We perform the construction of the graph structure and the construction of the node features respectively. A quadtree is essentially a graph, so we construct a graph for each trajectory, and the trajectory graphs constructed in this way capture the spatial relationships between trajectory points and reflect the differences in density in different regions.

Graph Structure Construction: Firstly, we perform the construction of the graph structure, which in turn involves the construction of graph nodes and edges.

The node set N contain not only the original trajectory nodes but also the parent nodes of the trajectory nodes, i.e., $N = N_{ori} \cup N_f$. Thus the graph G_T contains the hierarchical information of the trajectory. Specifically, N_{ori} is the original record point of a given trajectory T, $N_{ori} = \{n_1, n_2, \ldots, n_{len(T)}\}$. We traverse the nodes in N_{ori} sequentially and then extract the parent node of the original record point in $\mathcal{HQT}$, to get the list $N_f = [n_f^1, n_f^2, n_f^3, n_f^4, \ldots, n_f^{len(T)}]$, and then remove the duplicated nodes, to form the set of non-duplicated hierarchical nodes, e.g., $N_f = [n_f^1, n_f^3, n_f^4, \ldots, n_f^{len(T)}]$. And finally we get our node set $N = [n_1, n_2, \ldots, n_{len(T)}, n_f^1, n_f^3, n_f^4, \ldots, n_f^{len(T)}]$.

After constructing the nodes, let's construct the edges. We did not fully connect the nodes after we considered the memory efficiency of GPU and the computational cost of the model. We designed three different kinds of edges in G_T, all of which are undirected, which allows messages to be passed between nodes in both directions. The first are the edges between leaf nodes i.e., the original records of the trajectory, and we add fully connected edges between nodes in N_{ori}. The second are the edges between parent nodes i.e., hierarchical nodes, we add fully connected edges between nodes in N_{ori} as well. Finally, there are the edges between the hierarchical nodes and the original records, we add edges between the corresponding nodes in N_{ori} and N_f, e.g., between n_1 in N_{ori} and n_f^1 in N_f.

Node Feature Construction: We treat the trajectory graph G_T as homogeneous graph, i.e., we treat each node equally. Our node features consist of three aspects: geographic feature l_g, spatial hierarchical feature l_h, and region feature l_z. It is obvious that each node in N represents a location or a rectangular region in Zone $\mathcal{Z}$. For nodes in N_{ori}, we directly use the original recorded location as the information of the geographic feature. For the points in N_f, we choose their center positions as the information for the geographic features. We use a normalization function for the GPS coordinates in the trajectory and then transform the information of the nodes in N_{ori} and N_f with a multilayer perceptron. Assuming the location of n_i in N is (lat_i, lon_i), the geographic feature is constructed as:

$$l_{g(i)} = \omega_1(Normalize(lat_i, lon_i)).$$

We use the embedding matrix E_T of the $\mathcal{HQT}$ obtained in Section II to construct spatial hierarchical features:

$$l_{h(i)} = E_T(n_i),$$

where $n_i \in N$. This implies that connections are established between different trajectories whose original recorded locations are close in space by sharing similar features. Finally, modeling the size of a rectangular region in Zone requires the region feature l_z. If $n_i \in N_f$, we extract the length h_i and the width d_i of the associated rectangle, if $n_i \in N_{ori}$, we set h_i and d_i to 0, and then the length and width of the region size are also transformed by MLP:

$$l_{z(i)} = \omega_2(d_i, h_i).$$

After the above processing, for a particular node $n_i \in N$, we connect the above three parts together to get the node feature:

$$l_i = l_{g(i)} + l_{h(i)} + l_{z(i)}.$$

B. Route Inference

The trajectories are converted to graph structures and encoded through graph transformer and aggregation layer that reflect the complex spatial patterns in the trajectories. This part is designed to help the model learn the similarities and differences between different trajectories. Trajectory route inference is designed to learn and discover the distribution of normal routes by representing trajectories as vectors in the latent space for anomalous trajectory detection. Therefore, we propose a route inference network $q_\varphi(z|T)$ to infer the route representation vector for a given trajectory T. Here, φ is a parameter of $q_\varphi(z|T)$. In this paper, we use Graph Transformer to implement $q_\varphi(z|T)$, because it captures both the sequence information of variable-length trajectories and the spatial hierarchical information, and capturing this rich spatio-temporal pattern enables better inference. Graph Transformer still adopts the main idea of Transformer, which captures

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.

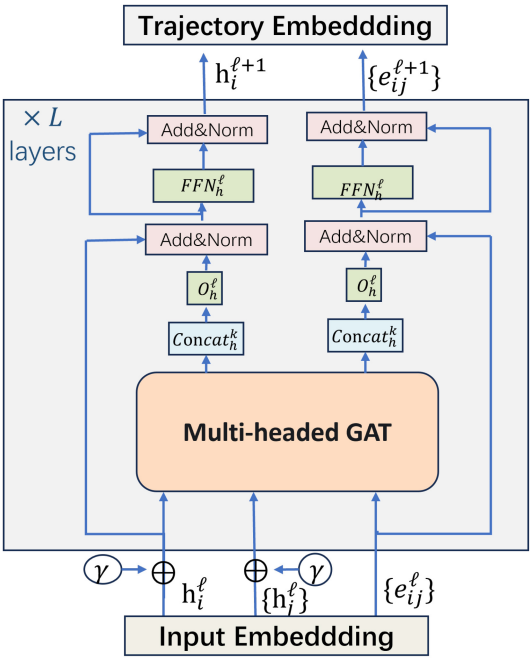


Fig. 4. Graph Transformer layer with edge features.

long trajectories' dependencies. It takes a sparse graph as input, and not every node needs to focus on other nodes, which reduces GPU memory consumption. Compared with the traditional Transformer model, our model has two novel designs: (1) position encoding is represented by Laplacian feature vectors. (2) replacing layer normalization with batch normalization layers. (3) extending this architecture to edge feature tables. We describe these designs below.

1) Laplacian Position Encoding: With Laplace position coding, we can model positional relationships for nodes in G_T by considering relative distance information. This provides an efficient way to capture the relationships in the graph by introducing different positional features for neighboring nodes and far away nodes, as shown in Equation (1):

$$\begin{cases} \Delta = I - D^{-1/2}AD^{-1/2} = U^T \Lambda U \\ \gamma_i^g = MLP(min_k(U)), \end{cases} \tag{1}$$

where A is the adjacency matrix, D is the degree matrix, and Λ, U correspond to the eigenvalues and eigenvectors, respectively. γ_i^g denotes the Laplace positional encoding of node i, and min_k denotes the k smallest nontrivial eigenvectors of the selected Λ. By feeding the feature vectors to the fully connected layer, we obtain the position encoding of the graph, denoted as $\gamma_i^g \in R^{d/2}$. We sum the position encoding vector with the node features: $g_i = \gamma_i + l_i$, we take g_i as the input to the next Graph Transformer Layer, where $g_i \in R^d$ is the input vector of node i.

2) Graph Transformer Layer: In order to capture the long-term distance dependence and spatial hierarchical relationships of trajectories, we employ graph transformer. At this point, the self-attention module in Transformer becomes a graph attention network. In order to better utilize the rich feature information of the trajectory graph, we introduce the edge attributes of the graph. We associate these available edge features with implicit edge scores computed through pairwise attention. For a particular node n_i, we take g_i as input h_i^0, and the operations in the Graph Transformer layer are defined as follows:

$$\begin{aligned} \widehat{h}_i^{l+1} &= O_h^l \; concat_{k=1}^{H} \left(\sum_{j \in N_i} w_{i\,j}^{k,l} V^{k,l} h_j^l \right), \\ \widehat{e}_{i\,j}^{l+1} &= O_e^l \; concat_{k=1}^{H} \left(\widehat{w}_{i\,j}^{k,l} \right), \\ where, \; w_{i,j}^{k,l} &= softmax_j \left(\widehat{w}_{i,j}^{k,l} \right), \\ \widehat{w}_{i,j}^{k,l} &= \left(\frac{Q^{k,\,l} h_i^l \cdot K^{k,\,l} h_j^l}{\sqrt{d_k}} \right) \cdot E^{k,\,l} e_{i,j}^l. \end{aligned} \tag{2}$$

where $Q^{k,l}, K^{k,l}, V^{k,l}, E^{k,l} \in \mathbb{R}^{d_k \times d}$, $O_h^l, O_e^l \in \mathbb{R}^{d \times d}$, $k = 1$ to H represents the number of attention heads. The output after we exponentiate the terms inside softmax is restricted to values between -5 and $+5$, which is for numerical stability. The outputs $\hat{h}_i^{l+1}$ and $\hat{e}_{i,j}^{l+1}$ are then passed to a feed-forward network (FFN) containing residual joins and batch normalization operations. For simplicity, we omit the equations for the same operations as the Transformer.

After going through the L layer of Graph Transformer we get the embedding vectors of the nodes of the trajectory graph, i.e., $[h_1^L, h_2^L, h_3^L, \ldots, h_N^L]$. Edge features are only used to change the attention score here, and we do not do anything with the edge features of node i, i.e., $e_{i,j}^{l+1}$. In order to obtain the latent space vector z of the trajectory, we use the mean readout function for the embedding of the nodes, which is Aggregation Layer. The final latent space z of the route can be computed by the following equation:

$$z = \sum_{i-1}^{N} h_i^L / N. \tag{3}$$

3) Gaussian Distribution Modeling: To cope with uncertainty and noise in trajectories, we model the probability distribution of vectors in the latent space using a Gaussian distribution. We use the polynomial distribution $p_\varphi(v) = Mult(\pi)$ to model the probability of a path traversed by a trajectory of a particular type (denoted v), with π as a parameter of the distribution. The latent vector z of a trajectory T can be expressed as:

$$z_T \sim p_\varphi(z|v) = N\left(\mu_v, \sigma_v^2\right), \tag{4}$$

where $p_\varphi(z|v)$ is a Gaussian distribution and denotes the probability that a trajectory travels through a route of type v. μ_v and σ_v^2 denote the mean and variance of a distribution of type v, which can be learned by a neural network $\omega_3(\cdot)$. After introducing the latent route type v, we need to extend the inference network $q_\varphi(z|T)$ to $q_\varphi(z, v|T)$ in order to infer the route type v in addition to the latent route z. Thus, we can apply the principle of the mean-field approximation to infer the latent vector z and the route type v from the following network:

$$q_\varphi(z, v|T) = q_\varphi(z|T) q_\varphi(v|T), \tag{5}$$

where $q_\varphi(z|T)$ obeys a Gaussian distribution $N(\mu_T, \sigma_T^2)$.We can consider $q_\varphi(v|T)$ to be the probability of route type v given

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.

the route of a trajectory T. It can be denoted by $p_\varphi(v|r_T)$, and thus it can be defined as:

$$q_\varphi(v|T) = p_\varphi(v|r_T) = \frac{p_\varphi(v)p_\varphi(z_T|v)}{\sum_{i=1}^{v} p_\varphi(i)p_\varphi(z_T|i)}. \quad (6)$$

Finally, the route inference network infers the route latent space z from a given trajectory. The learnable parameters in the route inference network include $\varphi = \{\omega_1(\cdot), \omega_2(\cdot), Q^{k,l}, K^{k,l}, V^{k,l}, E^{k,l}, \pi, \omega_3(\cdot)\}$.

C. Trajectory Generation

Generating trajectories through Gaussian distributions and GRU networks reflect an unbiased understanding of the spatial distribution of trajectories. The model generates new trajectories by learning the normal distribution. In the previous section, the inference network has learned the vectors in the route latent space, so the next step is to generate trajectories based on the vectors in the latent space. To achieve this goal, we design a new generative network $p_\theta(T|z)$, where θ is the set of learnable parameters. Our goal is to optimize $p_\theta(T|z)$ to generate trajectories well from its inferred route latent space z. For simplicity and to reduce computational cost, we use a GRU neural network as the generative network. For each step i, the generation result x_i is based on the latent route and the past sequences $x_{<i}$, so we recurrently encode z and the sequences $x_{<i}$ into a hidden vector y_i and use a multinomial distribution to generate our trajectory:

$$\begin{aligned} y_i &= \omega_4(x_i, y_{i-1}) \quad i = 1, 2 \ldots, n \quad y_0 = z, \\ x_i &\sim p_\theta(x_i|x_{<i}, z) = p_\theta(x_i|y_{i-1}) = Mult(\omega_5(y_{i-1})). \end{aligned} \quad (7)$$

$\theta = \{\omega_4(\cdot), \omega_5(\cdot)\}$ are learnable parameters, and the training process is shown in the following section. $\omega_5(\cdot)$ transforms y_{i-1} into our desired 2D vector, i.e., latitude and longitude. By doing this at each time step, it can loop through the generation of trajectory points and finally obtain our one whole trajectory.

D. Optimization

The parameters that can be learned in our model include $\varphi = \{\omega_1(\cdot), \omega_2(\cdot), Q^{k,l}, K^{k,l}, V^{k,l}, E^{k,l}, \pi, \omega_3(\cdot)\}$ and $\theta = \{\omega_4(\cdot), \omega_5(\cdot)\}$. The goal of our training is to jointly optimize the inference network and the generative network, so that the generative network can better generate trajectories. Specifically, it is to maximize the marginal log-likelihood of the generated training data, which is represented as follows:

$$\log p_\theta\left(T^{(1)}, T^{(2)}, \ldots, T^{(N)}\right) = \log p_\theta\left(\tau^{(1)}, \tau^{(2)}, \ldots, \tau^{(N)}\right), \quad (8)$$

where $T^{(1)}, T^{(2)}, \ldots, T^{(N)}$ are raw trajectory set in the training stage, $\tau^{(1)}, \tau^{(2)}, \ldots, \tau^{(N)}$ are generated trajectories set. The marginal log-likelihood is optimized by maximizing the Evidence Lower Bound (ELBO) [37] of the marginal log-likelihood, which is represented as follows:

$$\begin{aligned} \log p_\theta(T) &\geq ELBO = \mathcal{L}(\varphi, \theta; T) \\ &= \mathbb{E}_{p_\varphi(z,v|T)}\big[-\log p_\varphi(z, v|T) \\ &\quad + \log p_\theta(T, z, v)\big]. \end{aligned} \quad (9)$$

Algorithm 1 The Overall Procedure of HS-UATD

Input: Normal trajectory training set: T_{train}, Trajectory test set with anomalies: T_{test}, Training epochs: T, Batch size: B.

Output: Anomaly scoring *score* of trajectory test set.

1: Construct Hierarchical Quadtree and Trajectory Graph for T_{train} and T_{test}, as shown in Section IV-A.
2: Initialize the trainable parameters φ, θ randomly.
3: *// Training phase*
4: **for** t in $\{1, 2, \ldots, T\}$ **do**
5: **for** batch $(G_1, G_2, ..., G_B)$ **do**
6: Turning B graphs into a Batchgraph.
7: Get latent space vector z via Eq. (3).
8: Construct Gaussian distribution $p_\varphi(z|v)$ with μ_v, σ_v^2 via Eq. (4).
9: Construct $q_\varphi(v|T)$ via Eq. (6).
10: Calculate $q_\varphi(z, v|T)$ via Eq. (5).
11: Obtain generated T via Eq. (7).
12: Construct ELBO $= \mathcal{L}(\varphi, \theta; T)$ of the marginal log-likelihood via Eq. (9).
13: Optimize $\mathcal{L}(\varphi, \theta; T)$ to update the parameters φ, θ.
14: **end for**
15: **end for**
16: *// Test phase*
17: **for** $Tr_i \in T_{test}$ **do**
18: Obtain generated T via Eq. (7).
19: Calculate the probability of generating a normal trajectory *likelihood* via Eq. (10).
20: **return** anomaly *score* : $s(T) = 1 - likelihood$.
21: **end for**

Lines 3 to 15 of Algorithm 1 presents the process of HS-UATD model training.

E. Anomalous Trajectory Detection

We use a generative network to generate trajectory samples from a normal latent spatial route distribution, and the anomaly score is determined by measuring how much the trajectory deviates from the normal trajectory. Higher anomaly scores indicate that the trajectory differs significantly from normal behavior and may be an anomalous trajectory. This approach allows us to identify anomalous trajectories that deviate from the normal trajectory distribution during the generation phase, thus improving the accuracy of anomaly detection. $p_{\sigma(t_i|t_{<i},\mu)}$ in the Anomalous Trajectory Detection module of Fig. 2 refers to the probability that a trajectory point is a normal trajectory point and is used in the *likelihood* calculations. In this paper, the route type $v = 1$, so the anomaly score s for a generative trajectory T can be defined as follows:

$$likelihood = exp\left[\frac{\sum_{i=1}^{n} \log p_{\sigma(t_i|t_{<i},\mu)}}{n}\right], \quad (10)$$

$$s(T) = 1 - likelihood, \quad (11)$$

where n is the length of the trajectory, which is used to normalize the anomaly scores for trajectories of different

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.

TABLE I
DATASET TRAJECTORY CHARACTERISTICS

Dataset	Number	Minimum Time	Maximum Time	Minimum Distance	Maximum Distance
SD1	2631	280s	2850s	1850m	23782m
SD2	2283	276s	2205s	1998m	15024m
SD3	2214	345s	3210s	4596m	19797m

lengths. The *likelihood* is the value of the probability of generating a normal trajectory, and it ranges from -1 to 1. This part is implemented in lines 16 to 21 of Algorithm 1.

V. EXPERIMENTS

A. Experiment Settings

1) Datasets Processing: Our experiments are conducted on a real taxi trajectory dataset, which provided by Kaggle,[1] containing more than 1.71 million trajectory records generated by 442 taxicabs operating from January 7, 2013 to June 30, 2014 in Porto, Portugal. Each trajectory is a sequence consisting of a series of GPS coordinates reported by the taxi every 15 seconds. The trajectories are classified according to their (S,D) pairs, and we selected three (S,D) pairs with high traffic volume as datasets, which are named SD1, SD2, and SD3 for the convenience of the subsequent descriptions.

2) Datasets Analysis: In order to have a better understanding of the three datasets, we explore their trajectory characteristics as shown in Table I. And we compute the travel time and distance for each trajectory in the three original datasets at certain time intervals. It is worth noting that there are some anomalous trajectories in the datasets with travel times and distances beyond the typical range of normal trajectories. In order to ensure that the visualization results are reasonable, we eliminate those anomalous trajectories with large values before performing the visualization. Specifically, we remove the top 5% of trajectories with large values. In Fig. 5, we show the visualization results for the remaining 95% of the trajectory data, where the y-axis indicates the proportion of trajectory features in the value interval.

From the Fig. 5, it can be observed that the travel times and distances of the trajectories in SD1 and SD2 show a tendency to be concentrated in a specific range, and their statistical distributions are characterized by Gaussian distributions. In contrast, the travel time in SD3 is similar to the first two datasets, but the distance presents two distinct categories, implying the existence of two completely different travel routes in the dataset, which are far apart in distance but similar in time. In addition, according to Fig. 5, some of the trajectories in the three datasets clearly deviate from the normal range, which indicates the presence of abnormal trajectories. Since we want to model the probability distribution of normal trajectories, it is very necessary that we first perform data cleaning to get normal data.

3) Ground Truth: To obtain an unbiased training set and an unbiased testing set, we firstly clean the three (S,D) pairs by clustering algorithm to obtain normal trajectories, i.e., pure trajectories (PSD), and then divide the PSD into training and testing sets with a ratio of 4:1. Thus, unbiased training dataset is obtained. We inject temporal anomalies and detour anomalies into normal trajectories to obtain the unbiased testing set.

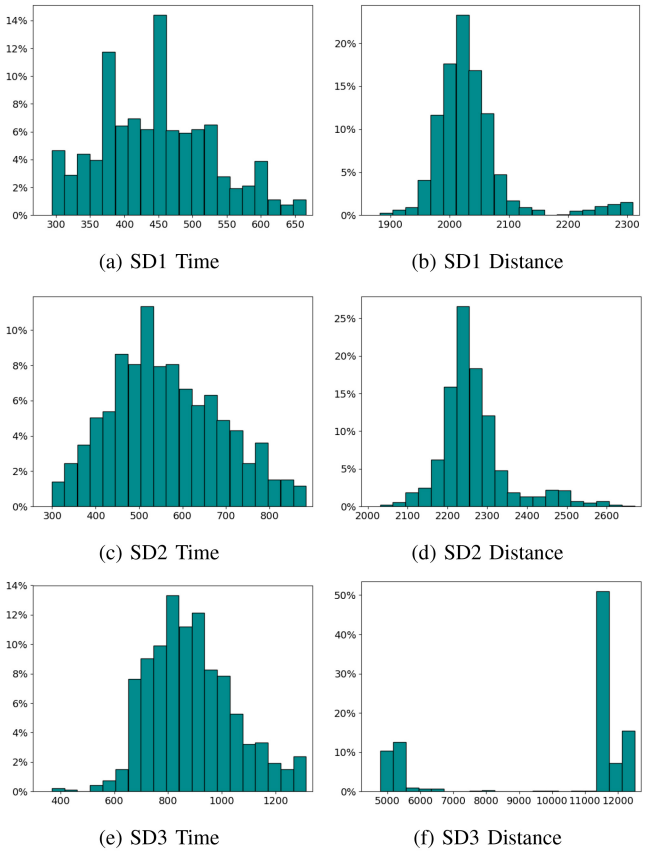


Fig. 5. Interval statistics of trajectory characteristics for three datasets.

Temporal anomalies. Such as abnormally deliberate slowing or speeding, We can obtain this by shortening or lengthening the trajectory duration. We use a quadratic interpolation technique to scale the shortest 4% of normal trajectories, reducing their duration to 50%-70% of the original time. At the same time, we lengthen the longest 1% of normal trajectories to increase their duration to 150%-200% of the original time. Such a ratio is set because lengthening trajectories is more likely to destroy their spatial properties than shortening them, and also generates spatial anomalies. Finally, we obtain the testing datasets injected with 5% temporal anomalies.

Detour anomalies. Such as spatial detours, we can obtain them by offsetting the trajectory segments. Specifically, 5% of the trajectories from the testing set are randomly selected, and 30% of the trajectory segments in each trajectory are randomly selected to be offset left and right in the direction of travel, but the offset segments cannot include the start and end points of the trajectories. Finally, we obtain the testing datasets injected with 5% detour anomalies.

In the next section, we refer to the original datasets as SD1, SD2, and SD3. And the processed datasets as PSD1, PSD2, and PSD3. LPSD1 refers to the temporal anomaly dataset

[1]https://kaggle.com/competitions/pkdd-15-predict-taxi-service-trajectory-i

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.

generated based on PSD1, and RPSD1 refers to the detour anomaly dataset generated based on PSD1. LSD1 refers to the temporal anomaly dataset generated based on SD1, and RSD1 refers to the detour anomaly dataset generated based on SD1.

4) Baseline: In order to validate the performance of our proposed model, our proposed model is compared with the following models for anomalous trajectory detection:

- TPRO [38]: Detecting anomalies by segmenting trajectories.
- iBAT [8]: This is an online detection framework. It detects anomalies by comparing the isolation degree of other trajectories, which share the same start and end points as the target trajectory.
- ATDC [9]: The method designs anomaly scoring functions to distinguish different types of anomalous trajectories, and utilizes set difference and intersection to evaluate the similarity between any two trajectories.
- GM-VSAE [18]: It is an RNN-based Gaussian hybrid variational sequence autoencoder. This model learns the probability distribution of normal trajectories. Based on this, model can generates the likelihood of a target trajectory belonging to a normal trajectory.
- UA-OATD [19]: It is an improved version of GM-VSAE. This version considers the neighbor information of the grid and introduces an attention mechanism to encode the trajectories.

5) Metrics: We use PR-AUC and F1-score as our metrics with scores ranging from 0-1, because they are well suited for evaluating the performance of anomaly detection methods on datasets with an imbalance of positive and negative samples, i.e., the anomalous trajectories generated are only a very small fraction of the dataset. And because our focus is on evaluating the model's ability to detect anomalous trajectories rather than categorize them, we choose PR-AUC and F1-score as the evaluation metrics. Since a customized threshold is required to derive the F1-score each time, we focus our discussion on PR-AUC.

6) Parameters Setting: The embedding dimension of our experiments is 32, threshold of trajectory record points contained in the leaf nodes η is 40, the sample size n is 10, and the graph transformer uses a three-layer structure with 8 attention heads per layer of the multi-head graph attention structure. The model uses Adam optimizer with a learning rate of 0.001. We have used the K-means clustering algorithm with detailed configurations: the number of clusters is set to 2, the initialization method is K-means++, the DTW is used as the similarity measure, the maximum number of iterations is 300, the convergence criterion is 1e-4, and the random seed is 42.

B. Effectiveness Evaluation

1) Unbiased Datasets:

Detour anomalies. We perform detour anomalous trajectory detection experiments on three datasets RPSD1, RPSD2, and RPSD3. The performance comparison of all the methods is shown on the left side of Table II.

For the analysis of results across methods, the results show that our proposed method performs very well on all three datasets. Compared with traditional methods such as iBAT and ATDC, the deep learning methods show significant improvement in processing trajectory data. The deep learning methods outperform the traditional methods by several times on all three datasets. Compared with the current best method UA-OATD, the proposed HS-UATD improves the PR-AUC score by 5.04% on average and the F1-score by 1.3% on average. UA-OATD outperforms GM-VSAE because it not only considers the grid's neighboring information but also introduces an attention mechanism. This mechanism allows for more efficient encoding of trajectories. HS-UATD achieves better performance than UA-OATD primarily due to its spatial hierarchical modeling. This modeling captures the spatial density distribution of trajectories and makes the captured trajectory information more comprehensive. Additionally, it alleviates some disadvantages of using equal-sized grids.

For the analysis of results across datasets, this improvement is especially significant in spatial detour anomaly detection. Our method achieves the highest performance on the RPSD1 dataset. This can be explained by the fact that the trajectory distribution is denser in the RPSD1 dataset. And by utilizing this dense trajectory data, the new method is able to better learn the spatial features of the trajectories and obtain better vectors in the latent space. On the contrary, other methods have difficulties in dealing with dense trajectory sets effectively due to inherent limitations, which affects their performance. TPRO detects anomalies by segmenting trajectories, which tends to overlook subtle spatial pattern differences in dense regions. IBAT relies on comparing the degree of isolation of trajectories with the same start and end points. However, this approach is prone to misclassification in complex and dense trajectory data. In such cases, trajectories with the same start and end points may have multiple normal variants. Although ATDC uses an anomaly scoring function to identify anomalous trajectories, its set difference and intersection-based evaluation cannot accurately capture small differences between trajectories when faced with densely distributed trajectories. Our method exhibits poor performance on the RPSD3 dataset containing detour anomalies compared to other datasets. The SD3 dataset contains two completely different typical routes. These routes are similar in time but differ significantly in distance, making it challenging to process clustering-labeled anomalous trajectories. The reason is that clustering algorithms usually rely on the temporal information and spatial features of trajectories to distinguish normal trajectories from abnormal ones. The two trajectories are similar in time, thus algorithms may mistakenly consider them as the same type of trajectory, ignoring their significant differences in spatial distance. As a result, some abnormal trajectories are mislabeled as normal by the clustering algorithm and some normal trajectories are mislabeled as abnormal, leading to misclassification of trajectories in the RPSD3 dataset. Overall, HS-UATD is able to detect spatial detour anomalous trajectories.

Temporal anomalies. We perform temporal anomaly trajectory detection experiments on three datasets LPSD1, LPSD2, and LPSD3. The performance comparisons of all the methods are shown on the right-hand side of Table II, which shows that our proposed method performs very well on all three datasets.

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.

TABLE II
PERFORMANCE COMPARISON ON UNBIASED DATASET

Dataset	Detour Anomalies						Temporal Anomalies					
	RPSD1		RPSD2		RPSD3		LPSD1		LPSD2		LPSD3	
Metrics	F1	PR-AUC	F1	PR-AUC	F1	PR-AUC	F1	PR-AUC	F1	PR-AUC	F1	PR-AUC
TPRO	0.210	0.305	0.245	0.235	0.264	0.300	0.224	0.330	0.167	0.224	0.278	0.318
IBAT	0.305	0.400	0.264	0.304	0.208	0.284	0.185	0.167	0.111	0.128	0.254	0.383
ATDC	0.364	0.445	0.520	0.467	0.386	0.432	0.258	0.368	0.354	0.303	0.164	0.224
GM-VSAE	0.910	0.940	0.803	0.827	0.730	0.778	0.885	0.880	0.610	0.532	0.710	0.683
UA-OATD	0.939	0.947	0.904	0.922	0.850	0.887	**0.946**	**0.920**	0.724	0.680	0.860	0.667
HS-UATD	**0.967**	**0.987**	**0.934**	**0.962**	**0.925**	**0.947**	0.862	0.820	**0.729**	**0.748**	**0.871**	**0.812**

TABLE III
PERFORMANCE COMPARISON ON BIASED DATASET

Dataset	Detour Anomalies						Temporal Anomalies					
	RSD1		RSD2		RSD3		LSD1		LSD2		LSD3	
Metrics	F1	PR-AUC	F1	PR-AUC	F1	PR-AUC	F1	PR-AUC	F1	PR-AUC	F1	PR-AUC
TPRO	0.195	0.255	0.197	0.204	0.153	0.265	0.137	0.236	0.105	0.147	0.127	0.163
IBAT	0.269	0.387	0.195	0.217	0.154	0.273	0.153	0.167	0.102	0.158	0.103	0.187
ATDC	0.350	0.425	0.312	0.320	0.259	**0.375**	0.141	0.167	0.134	0.117	0.134	0.138
GM-VSAE	0.850	0.630	0.667	0.421	0.491	0.250	0.710	0.432	0.614	0.469	0.497	0.283
UA-OATD	**0.861**	0.657	0.710	0.456	0.562	0.314	0.735	0.526	**0.689**	0.442	0.585	0.296
HS-UATD	0.763	**0.706**	**0.747**	**0.555**	**0.567**	0.289	**0.857**	**0.645**	0.671	**0.571**	**0.642**	**0.559**

The results are similar to those of detour anomaly detection. Compared with the state-of-the-art UA-OATD method, HS-UATD achieves the best results in both LPSD2 and LPSD3, and our method improves the PR-AUC score by 10.0% in the LPSD2 dataset and 21.74% in the LPSD3 dataset. UA-OATD and GM-VSAE are ranked second and third. However, the overall detection performance is poorer than the detection of anomalous trajectories at the detour. This is because the anomalies generated by quadratic interpolation scaling are weaker spatially and more pronounced in dense trajectories. As a result, UA-OATD replaces our method to achieve the best performance in LPSD1. Our method shows better performance on the LPSD3 dataset than the other datasets. This can be explained by the fact that the trajectories in LPSD3 cover longer distances than those in other datasets, and there are two trajectories with a large spatial disparity. When scaling using quadratic interpolation, the spatial properties of these trajectories are somewhat disrupted. Overall, our method achieves excellent results on all three datasets, validating its ability to detect temporal anomalous trajectories.

2) Biased Datasets: We perform temporal and detour anomalous trajectory detection experiments on three unprocessed biased raw datasets SD1, SD2, and SD3. And the performance comparison of all methods is shown in Table III. All methods that require learning the probability distribution of the normal trajectories show a substantial performance degradation compared to the previous experiments. All methods perform best on the RSD1 dataset containing detour anomalies, so it is used to measure the pure dataset's importance. GM-VASE shows a 32.98% performance drop compared to detection in the pure dataset, UA-OATD shows a 30.62% performance drop, and our model shows a 28.47% performance drop. These results indicate that using clean, unbiased training datasets improves detection performance. The experiment confirms that the original dataset is biased, and processing the raw data is essential for improving detection accuracy. This supports the rationale behind our approach. The results from the overall experiment show that our proposed method performs remarkably well on all three datasets. The results are similar to the previous tests.

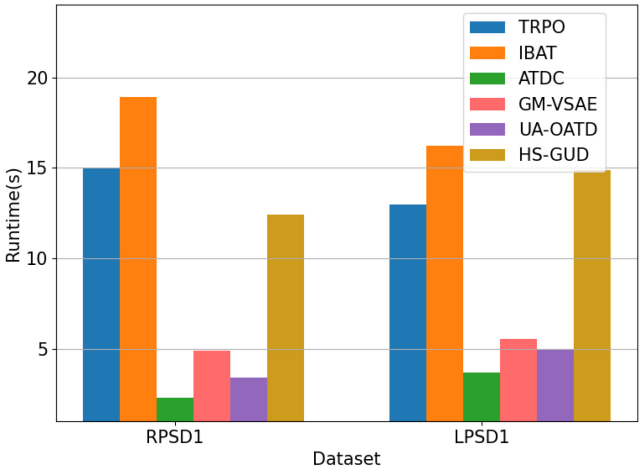
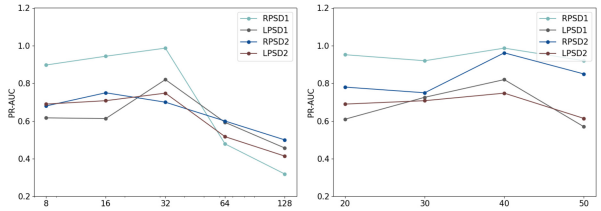


Fig. 6. Running time of each method on RPSD1 dataset.

C. Efficiency Evaluation

We conduct a study on the efficiency of anomaly detection methods. We perform a comparison of the detection time of six methods on the RPSD1 dataset. The results show that the deep learning-based methods exhibit shorter detection time than the other methods in general. Specifically, GM-VSAE and UA-OATD have comparable detection time,

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.



(a) Embedding size d (b) Records number threshold η

Fig. 7. Parameter performance comparison of HS-UATD.

TABLE IV
ABLATION STUDY ON UNBIASED DATASET

PR-AUC Method \ Dataset	RPSD1	RPSD2	RPSD3
HS-UATD-H	0.945	0.927	0.912
HS-UATD-Graph	0.911	0.925	0.875
HS-UATD-Trans	0.956	0.916	0.926
HS-UATD	**0.987**	**0.962**	**0.947**

while HS-UATD has approximately three times the detection time of UA-OATD. This difference mainly comes from the data preprocessing stage. Specifically, hierarchical sequence modeling converts raw trajectories into trajectory maps, which takes some time. The other two parts of data preprocessing: Hierarchical Quadtree Construction and node embedding in quadtrees, do not affect our detection time. However, the two parts take up some time in the training phase. Therefore, improving runtime is a future research direction.

D. Parameter Study

This section investigates the sensitivity of hyperparameters and explores the model key parameters including the threshold of trajectory record points contained in the leaf nodes η, and the embedding size of the neural network d. To explore the impact of these parameters, we conduct experiments on the PSD1 and PSD2 datasets with different settings for each parameter.

Under a certain region $\mathcal{Z}$, the size of η affects the construction of the quadtree, and different η leads to quadtrees with different depths, which also affects the detection performance. As shown in the Fig. 7(b), as η increases, the performance of TrajGAT increases and then decreases. At $\eta = 40$, HS-UATD has the best performance, indicating that this threshold better captures the hierarchical structure information of the trajectory.

As shown in the Fig. 7(a), the best performance is achieved when d=32, and the model does not achieve the best performance when the embedding dimension is less than 32, indicating that smaller embedding dimension learns limited information. When the embedding dimension increases, the learned information is more but the model may suffer from overfitting.

E. Ablation Experiment

Our model employs a hierarchical structure modeling based on quadtree and graph transformer for anomalous trajectory detection, and we design a set of ablation experiments to verify the effectiveness of these two modules in our model.

- HS-UATD-H means that no hierarchical structure modeling based on quadtree is employed, the spatial region is divided into equal-sized grids, and graph transformer is still used for trajectory inference.
- HS-UATD-Graph does not construct a graph, uses leaf nodes of a quadtree as spatial partitions, and uses transformer for trajectory inference.
- HS-UATD-Trans refers to constructing the graph and then using only the graph attention network for trajectory inference.

In the absence of hierarchical structure modeling, the model's performance degrades by 4.3%, 3.6%, and 3.7% in the three datasets, respectively. Additionally, in the absence of graph structure and Transformer, the model's performance in the three datasets is reduced by an average of 6.4% and 3.4%, respectively. These experiments demonstrate the effectiveness of our proposed model. The hierarchical structure modeling, based on quadtree and graph construction, captures the spatial structure of trajectories more effectively. The Transformer model models trajectory sequence dependencies well. Together, these approaches improve anomalous trajectory detection, with our model achieving the best performance.

F. Case Study

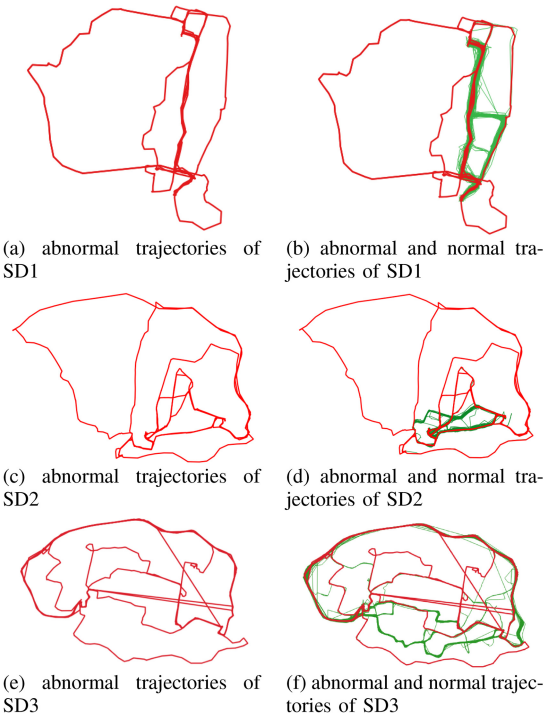
We test and visualize them in three test datasets, i.e., SD1, SD2, SD3, that are not injected with anomalies. We select ten trajectories with anomaly scores in the top ten and label them as anomalous, as shown in the Fig. 8(a), Fig. 8(c), Fig. 8(e). It is obvious that these trajectories are spatially different from the others, which meets the definition of detour anomalous trajectories, proving our method's effectiveness in detecting anomalous trajectories at the detour.

We validate the effectiveness of our approach in detecting temporal anomalies in the SD2 dataset. As shown in Fig. 9(a), the red-marked anomalous trajectory has an anomalous duration at the beginning of the sub-section, which is a slow-speed phenomenon. The vehicles in this dataset report the coordinates once in 15 seconds. The anomalous area has three straight-line segments, which indicates that it takes 45 seconds to pass through, whereas the other normal trajectories can pass through quickly. As shown in Fig. 9(b), the vehicle that produces the anomalous trajectory moves faster than other normal trajectories, which is a speeding phenomenon. Thus it proves the effectiveness of our method in detecting temporal abnormal trajectories.

VI. CHALLENGE AND FUTURE DIRECTIONS

In recent years, the detection of anomalous trajectories has faced many challenges such as, privacy security, isolated data island, insufficient computing power of vehicles and real-time traffic anomaly detection. In the following we describe one of the most important challenge and future directions for real-time anomaly trajectory detection.

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.

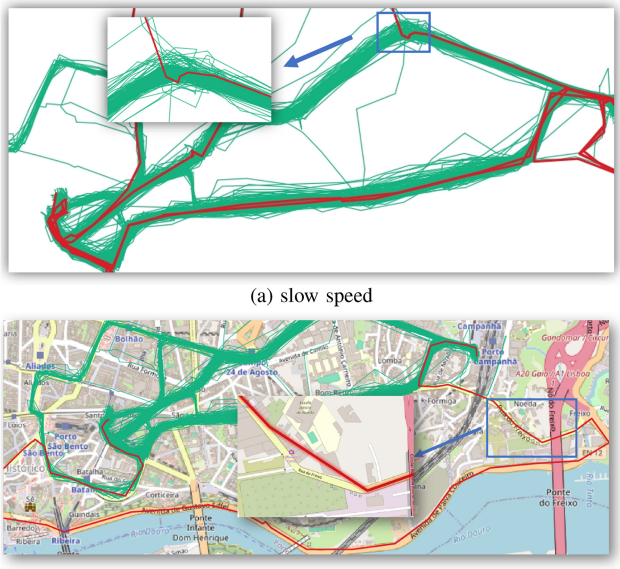


(a) abnormal trajectories of SD1 (b) abnormal and normal trajectories of SD1

(c) abnormal trajectories of SD2 (d) abnormal and normal trajectories of SD2

(e) abnormal trajectories of SD3 (f) abnormal and normal trajectories of SD3

Fig. 8. Visualization of detour anomaly detection results on three dataset.



(a) slow speed

(b) speeding

Fig. 9. Visualization of temporal anomaly detection results on SD2 dataset.

Existing real-time traffic anomaly detection faces multiple challenges such as computational complexity, data heterogeneity, privacy protection, and network communication latency. In recent years, Federated Learning Architecture and Edge Computing have developed rapidly. These solutions address privacy concerns and data transmission latency. And they improve real-time performance through local model training and edge device deployment. Distributed training, utilization of hardware gas pedals, and reduction of algorithmic time complexity have further optimized the hardware and performance of the system.

Future endeavors include further enhancing the generality of the model such as improving the temporal anomaly detection performance, and further investigating efficient algorithms such as improving the time efficiency of HS-UATD. Then distributed computing and high-performance platforms will be used to build an efficient real-time anomaly detection system that is more suitable for complex and dynamic urban transportation environments.

VII. Conclusion

In this paper, we propose a unbiased anomalous trajectory detection method (HS-UATD) based on hierarchical sequence modeling. By combining hierarchical modeling, graph encoding, and Gaussian distribution modeling, the model successfully captures the rich spatio-temporal patterns of trajectory data containing trajectory spatial density distribution. Our model is extensively experimented on three unbiased, biased and real trajectory datasets, and the results demonstrate the importance of the unbiased datasets and the superiority of HS-UATD in effectively detecting anomalous trajectories. The experiments validate the feasibility of the model in real-world scenarios such as identifying drivers traveling at inappropriate speeds and detecting taxi fraud. In addition, we provide insights into the challenge of real-time anomalous trajectory detection and provide an outlook on future research directions.

References

[1] D. Singh, S. P. Singh, and M. M. Al Dabel, "A directed graph and GRUs based trajectory forecasting of intelligent and automated transportation system for consumer electronics," *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 1601–1609, Feb. 2024.

[2] J. Jiang, D. Pan, H. Ren, X. Jiang, C. Li, and J. Wang, "Self-supervised trajectory representation learning with temporal regularities and travel semantics," in *Proc. IEEE 39th Int. Conf. Data Eng. (ICDE)*, 2023, pp. 843–855.

[3] P. Du, Y. Shi, H. Cao, S. Garg, M. Alrashoud, and P. K. Shukla, "AI-Enabled trajectory optimization of logistics UAVs with wind impacts in smart cities," *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 3885–3897, Feb. 2024.

[4] X. Kong, W. Zhou, G. Shen, W. Zhang, N. Liu, and Y. Yang, "Dynamic graph convolutional recurrent imputation network for spatiotemporal traffic missing data," *Knowl. Based Syst.*, vol. 261, Feb. 2023, Art. no. 110188.

[5] X. Kong, Q. Chen, M. Hou, H. Wang, and F. Xia, "Mobility trajectory generation: A survey," *Artif. Intell. Rev.*, vol. 56, no. 3, pp. 3057–3098, 2023.

[6] L. Sun, T. Ye, J. Sun, X. Duan, and Y. Luo, "Density-peak-based overlapping community detection algorithm," *IEEE Trans. Comput. Soc. Syst.*, vol. 9, no. 4, pp. 1211–1223, Aug. 2022.

[7] Q. Yu, F. Hu, Z. Ye, C. Chen, L. Sun, and Y. Luo, "High-frequency trajectory map matching algorithm based on road network topology," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 17530–17545, Oct. 2022.

[8] D. Zhang, N. Li, Z.-H. Zhou, C. Chen, L. Sun, and S. Li, "IBAT: Detecting anomalous taxi trajectories from GPS traces," in *Proc. 13th Int. Conf. Ubiquitous Comput.*, New York, NY, USA, 2011, pp. 99–108.

[9] J. Wang et al., "Anomalous trajectory detection and classification based on difference and intersection set distance," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 2487–2500, Mar. 2020.

[10] J.-G. Lee, J. Han, and X. Li, "Trajectory outlier detection: A partition-and-detect framework," in *Proc. IEEE 24th Int. Conf. Data Eng.*, 2008, pp. 140–149.

[11] X. Kong et al., "Spatial-temporal-cost combination based taxi driving fraud detection for collaborative Internet of Vehicles," *IEEE Trans. Ind. Informat.*, vol. 18, no. 5, pp. 3426–3436, May 2022.

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.

[12] D. Xia et al., "An ASM-CF model for anomalous trajectory detection with mobile trajectory big data," *Physica A, Stat. Mech. Appl.*, vol. 621, Jul. 2023, Art. no. 128770.
[13] J. Yang, X. Tan, and S. Rahardja, "MiPo: How to detect trajectory outliers with tabular outlier detectors," *Remote Sens.*, vol. 14, no. 21, p. 5394, 2022.
[14] J. Zhu, W. Jiang, A. Liu, G. Liu, and L. Zhao, "Time-dependent popular routes based trajectory outlier detection," in *Proc. Web Inf. Syst. Eng.*, 2015, pp. 16–30.
[15] Q. Zhang et al., "Online anomalous subtrajectory detection on road networks with deep reinforcement learning," in *Proc. IEEE 39th Int. Conf. Data Eng. (ICDE)*, 2023, pp. 246–258.
[16] K. K. Santhosh, D. P. Dogra, P. P. Roy, and A. Mitra, "Vehicular trajectory classification and traffic anomaly detection in videos using a hybrid CNN-VAE architecture," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 11891–11902, Aug. 2022.
[17] Y. Cheng, B. Wu, L. Song, and C. Shi, "Spatial-temporal recurrent neural network for anomalous trajectories detection," in *Proc. Int. Conf. Adv. Data Min. Appl.*, 2019, pp. 565–578.
[18] Y. Liu, K. Zhao, G. Cong, and Z. Bao, "Online anomalous trajectory detection with deep generative sequence modeling," in *Proc. IEEE 36th Int. Conf. Data Eng. (ICDE)*, Dallas, TX, USA, 2020, pp. 949–960.
[19] C. Wang, K. Li, and L. Chen, "Deep unified attention-based sequence modeling for online anomalous trajectory detection," *Future Gener. Comput. Syst.*, vol. 144, pp. 1–11, Jul. 2023.
[20] X. Kong, J. Wang, Z. Hu, Y. He, X. Zhao, and G. Shen, "Mobile trajectory anomaly detection: Taxonomy, methodology, challenges, and directions," *IEEE Internet Things J.*, vol. 11, no. 11, pp. 19210–19231, Jun. 2024.
[21] X. Han, R. Cheng, C. Ma, and T. Grubenmann, "DeepTEA: Effective and efficient online time-dependent trajectory outlier detection," *Proc. VLDB Endow.*, vol. 15, no. 7, pp. 1493–1505, 2022.
[22] Y. Zhang, N. Ning, P. Zhou, and B. Wu, "UT-ATD: Universal transformer for anomalous trajectory detection by embedding trajectory information," in *Proc. DMSVIVA*, 2021, pp. 70–77.
[23] H. Wu, W. Sun, and B. Zheng, "A fast trajectory outlier detection approach via driving behavior modeling," in *Proc. ACM Conf. Inf. Knowl. Manag.*, 2017, pp. 837–846.
[24] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "LSTM-based encoder-decoder for multi-sensor anomaly detection," 2016, *arXiv:1607.00148*.
[25] L. Deng, D. Lian, Z. Huang, and E. Chen, "Graph convolutional adversarial networks for spatiotemporal anomaly detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 6, pp. 2416–2428, Jun. 2022.
[26] J. Wyatt, A. Leach, S. M. Schmon, and C. G. Willcocks, "AnoDDPM: Anomaly detection with denoising diffusion probabilistic models using simplex noise," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 650–656.
[27] A. Mousakhan, T. Brox, and J. Tayyub, "Anomaly detection with conditioned denoising diffusion models," 2023, *arXiv:2305.15956*.
[28] C. Xiao, Z. Gou, W. Tai, K. Zhang, and F. Zhou, "Imputation-based time-series anomaly detection with conditional weight-incremental diffusion models," in *Proc. 29th ACM SIGKDD Conf. Knowl. Discov. Data Min.*, 2023, pp. 2742–2751.
[29] L. Song, R. Wang, D. Xiao, X. Han, Y. Cai, and C. Shi, "Anomalous trajectory detection using recurrent neural network," in *Proc. 14th Int. Conf. Adv. Data Min. Appl.*, 2018, pp. 263–277.
[30] D. Cai and W. Lam, "Graph transformer for graph-to-sequence learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 7464–7471.
[31] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous graph transformer," in *Proc. Web Conf.*, 2020, pp. 2704–2710.
[32] Y. Li, X. Liang, Z. Hu, Y. Chen, and E. P. Xing, "Graph transformer," in *Proc. ICLR*, 2019, pp. 1–14.
[33] V. P. Dwivedi, C. K. Joshi, A. T. Luu, T. Laurent, Y. Bengio, and X. Bresson, "Benchmarking graph neural networks," *J. Mach. Learn. Res.*, vol. 24, no. 43, pp. 1–48, 2023.
[34] J. Zhang, H. Zhang, C. Xia, and L. Sun, "Graph-bert: Only attention is needed for learning graph representations," 2020, *arXiv:2001.05140*.
[35] V. P. Dwivedi and X. Bresson, "A generalization of transformer networks to graphs," 2020, *arXiv:2012.09699*.
[36] H. Samet, "An overview of quadtrees, octrees, and related hierarchical data structures," in *Proc. Theor. Found. Comput. Graph. CAD*, 1988, pp. 51–68.
[37] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2013, *arXiv:1312.6114*.
[38] J. Zhu, W. Jiang, A. Liu, G. Liu, and L. Zhao, "Effective and efficient trajectory outlier detection based on time-dependent popular route," *World Wide Web*, vol. 20, no. 1, pp. 111–134, 2017.

Xiangjie Kong (Senior Member, IEEE) received the B.Sc. and Ph.D. degrees from Zhejiang University, Hangzhou, China, in 2004 and 2009, respectively, where he is currently a Full Professor with College of Computer Science and Technology. Previously, he was an Associate Professor with the School of Software, Dalian University of Technology, China. He has published over 200 scientific papers in international journals and conferences (with over 170 indexed by ISI SCIE). His research interests include network science, knowledge discovery, and urban computing. He is Distinguished Member of CCF and a member of ACM.

Yuwei He received the B.Sc. degree from China Jiliang University, Hangzhou, China, in 2022. She is currently pursuing the master's degree with the Zhejiang University of Technology, Hangzhou. Her main research interests include urban computing and data mining.

Guojiang Shen received the B.S. degree in control theory and control engineering and the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 1999 and 2004, respectively. He is currently a Professor with the College of Computer Science and Technology, Zhejiang University of Technology. His current research interests include artificial intelligence, big data analytics, and intelligent transportation systems.

Jiaxin Du received the Ph.D. degree from the School of Software, Dalian University of Technology, China, in 2023. She is currently a Lecturer with the College of Computer Science and Technology, Zhejiang University of Technology, China. Her current research interests include edge intelligence and security for wireless sensor networks.

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.

Zhi Liu received the B.S. degree in automatic control and the M.S. degree in system engineering from Xi'an Jiaotong University, Xi'an, China, in 1991 and 1994, respectively, and the Ph.D. degree in computer science and technology from Zhejiang University, Hangzhou, China, in 2001. She is currently a Professor with the College of Computer Science and Technology, Zhejiang University of Technology. Her current main research interests include intelligent transportation system and intelligent computing. She is a member of the China Computer Federation.

Ivan Lee received the B.Eng., M.Com., MER, and Ph.D. degrees from The University of Sydney. He had worked with Cisco Systems, Remotek Corporation, and Ryerson University. He was awarded a Researcher Exchange and Development within Industry Fellowship. He is currently an Associate Professor with the University of South Australia. His research interests include intelligent sensors, multimedia systems, and data analytics.

Authorized licensed use limited to: Zhejiang University of Technology. Downloaded on June 27,2025 at 08:12:26 UTC from IEEE Xplore. Restrictions apply.